Unsupervised Lab

Genevera I. Allen, Yufeng Liu, Hui Shen, Camille Little

7/22/2020

Data Description

gdat is Gene Expression Data, n = 445 patients x p = 353 genes

- Only 353 genes with somatic mutations from COSMIC are retained ## Data is Level III TCGA BRCA RNA-Sequencing gene expression data that have already been pre-processed according to the following steps:
- Reads normalized by RPKM
- Corrected for overdispersion by a log-transformation (1 + data)
- Short gene name labels are given as the column names

cdat is Clinical Data, n = 445 patients x q = 6 clinical features

- Subtype - denotes 5 PAM50 subtypes including Basal-like, Luminal A, Luminal B, HER2-enriched, and Normal-like
- ER-Status - estrogen-receptor status
- PR-Status - progesterone-receptor status
- HER2-Status - human epidermal growth factor receptor 2 status
- Node - number of lymph nodes involved
- Metastasis - indicator for whether the cancer has metastasized

Problems

Problem 1 - Dimension reduction

1a - Apply PCA, NMF, ICA and MDS, UMAP, and tSNE to this dataset. Compare and contrast the results using these methods.
1b - Relate the dimension reduction results with the clinical data. Is any clinical information reflected in the lower dimensional spaces?

1c - Overall, which dimension reduction method do you recommend for this data set and why?

Problem 2 - Clustering

2a - Apply various clustering algorithms such as K-means (explore different K), hierarchical clustering (explore different linkages), NMF, and biclustering. Compare the clustering results using these methods.

2b - Relate the clustering results with the clinical data. Can the clustering algorithm recover some of the clinical information such as cancer subtypes?

2c - Use Consensus Clustering to help Validate Clustering Results

2c - Overall, which clustering method(s) do you recommend for this data set and why?

Problem 3 - Multiple comparisons

3a - Identify important genes to differetiate ER postive and negative, PR postive and negative, HER2 postive and negative, metastasis status.

3b - Try different procedures to adjust for multiple comparisons.

3c - Examine the lists of genes identified using different methods for each clinical response. Which method is best? Why?

Problem 5 - Graphical models

5a - Use graphical models to explore interactions among genes. Are any of the well-connected genes related to patterns previously identified?

Problem 6 - Visulaization

6a - Visualize this data using multiple approaches.

6b - Prepare the "best" visual summary of this data.

Problem 7 - Exploratory Data Analysis Summary

7a - What is the most interesting finding?

7b - Is this finding consistent and stable?

7c - Prepare a visual summary that best illustrates this interesting finding.

R scripts to help out with the BRCA case study Lab Don't peek at this if you want to practice coding on your own!!

Load Data

```
load("UnsupL_SISBID_2020.Rdata")
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.6.2
```

```
library(ConsensusClusterPlus)
library(kknn)
library(GGally)
```

```
## Warning: package 'GGally' was built under R version 3.6.2
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
library(umap)
```

```
## Warning: package 'umap' was built under R version 3.6.2
```

```
library(Rtsne)
library(igraph)
```

```
##
## Attaching package: 'igraph'
```

```
## The following objects are masked from 'package:stats':
##
##     decompose, spectrum
## The following object is masked from 'package:base':
##
##     union
```

```
library(XMRF)
library(spacejam)
```

```
## Loading required package: splines
```

```
## Loading required package: Matrix
```

Explore Data

```
dim(gdat)
```

```
## [1] 445 353
```

```
dim(cdat)
```

```
## [1] 445   6
```

```
table(cdat$Subtype)
```

```
##
##    Basal-like HER2-enriched     Luminal A     Luminal B   Normal-like
##            79            53           200           106             7
```

```
table(cdat$ER)
```

```
##
##               Indeterminate                    Negative
##                           2                         100
##               Not Performed Performed but Not Available
##                           2                           2
##                    Positive
##                         339
```

```
table(cdat$PR)
```

```
##
##               Indeterminate                    Negative
##                           3                         147
##               Not Performed Performed but Not Available
##                           2                           2
##                    Positive
##                         291
```

```
table(cdat$HER2)
```

```
##
##     Equivocal      Negative Not Available      Positive
##             5           370             4            65
```

```
table(cdat$Node)
```

```
##
##   0   1   2   3
## 221 146  54  23
```

```
table(cdat$Metastasis)
```

```
##
```

```
##   0   1
## 427  11
```

```
table(cdat$ER,cdat$PR)
```

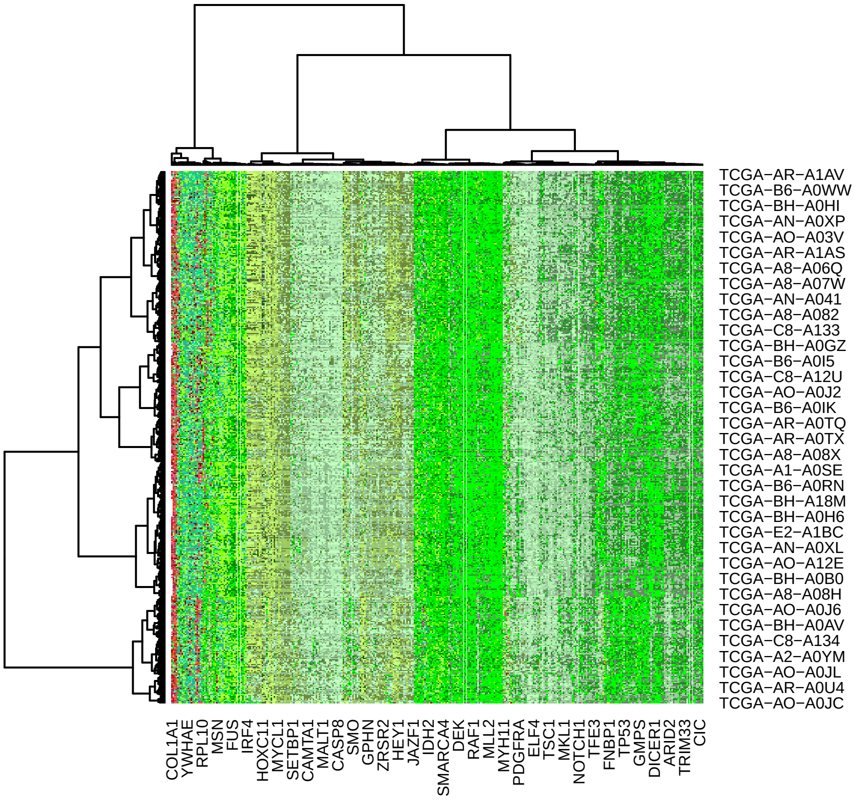
```
##
##                              Indeterminate Negative Not Performed
##   Indeterminate                          0        1             0
##   Negative                               1       93             0
##   Not Performed                          0        0             2
##   Performed but Not Available            0        0             0
##   Positive                               2       53             0
##
##                              Performed but Not Available Positive
##   Indeterminate                                        0        1
##   Negative                                             0        6
##   Not Performed                                        0        0
##   Performed but Not Available                          2        0
##   Positive                                             0      284
```

#cluster heatmap - biclustering

```
#cluster heatmap - biclustering
aa = grep("grey",colors())
bb = grep("green",colors())
cc = grep("red",colors())
gcol2 = colors()[c(aa[1:2],bb[1:25],cc[1:50])]
```

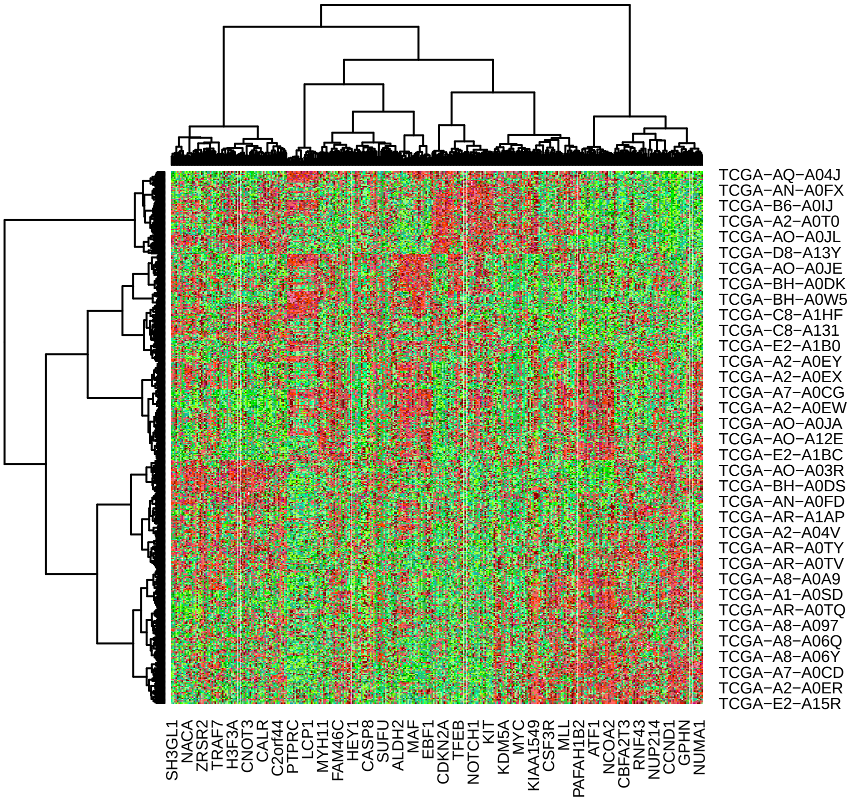
Without scaling

```
heatmap(gdat,col=gcol2,hclustfun=function(x)hclust(x,method="ward.D"))
```

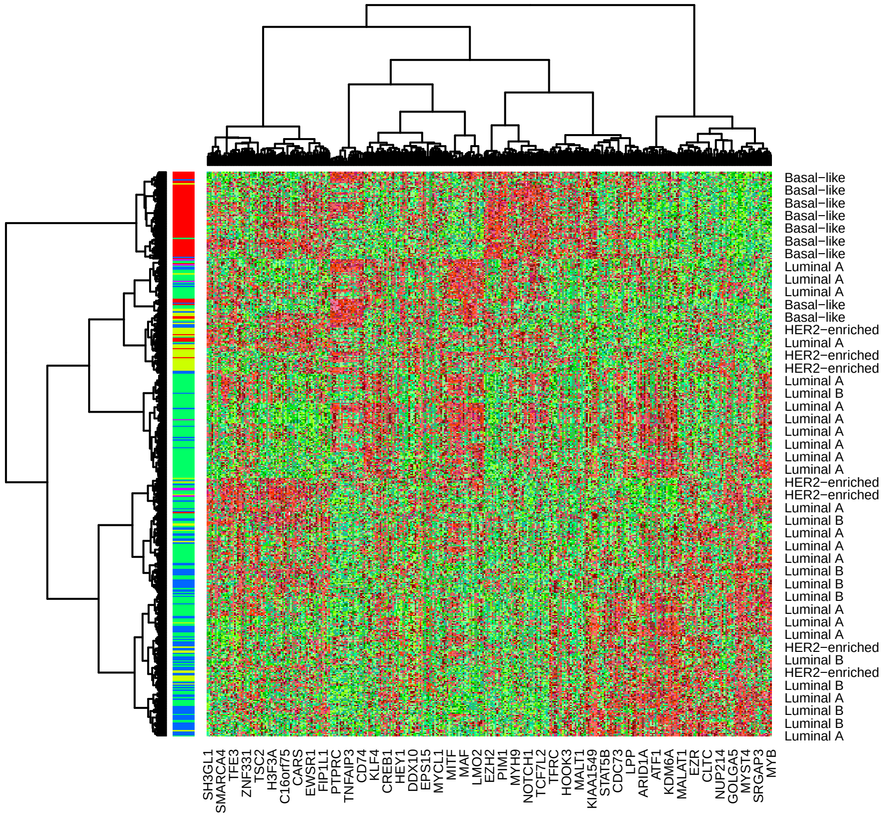


With scaling

```
heatmap(scale(gdat),col=gcol2,hclustfun=function(x)hclust(x,method="ward.D"))
```



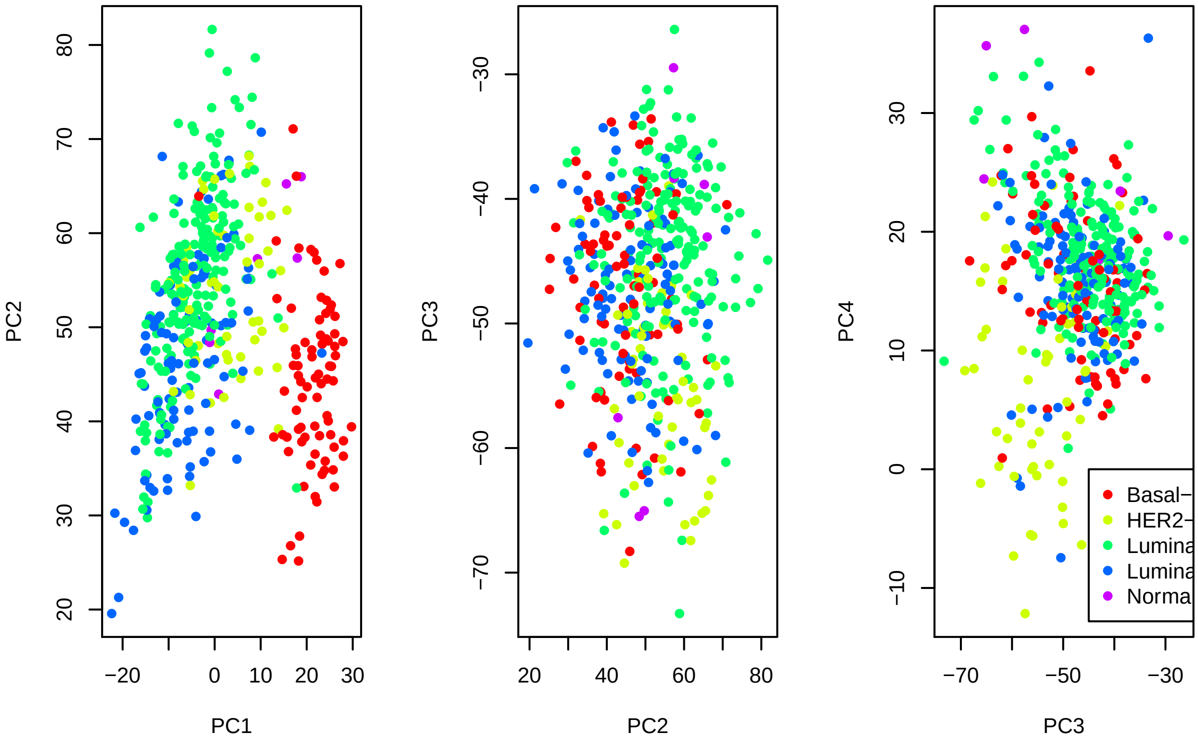
```
Cols=function(vec){cols=rainbow(length(unique(vec)))
                    return(cols[as.numeric(as.factor(vec))])}

heatmap(scale(gdat),col=gcol2,hclustfun=function(x)hclust(x,method="ward.D"),labRow=cdat$Subtype,RowSid
```



#Dimension Reduction

PCA

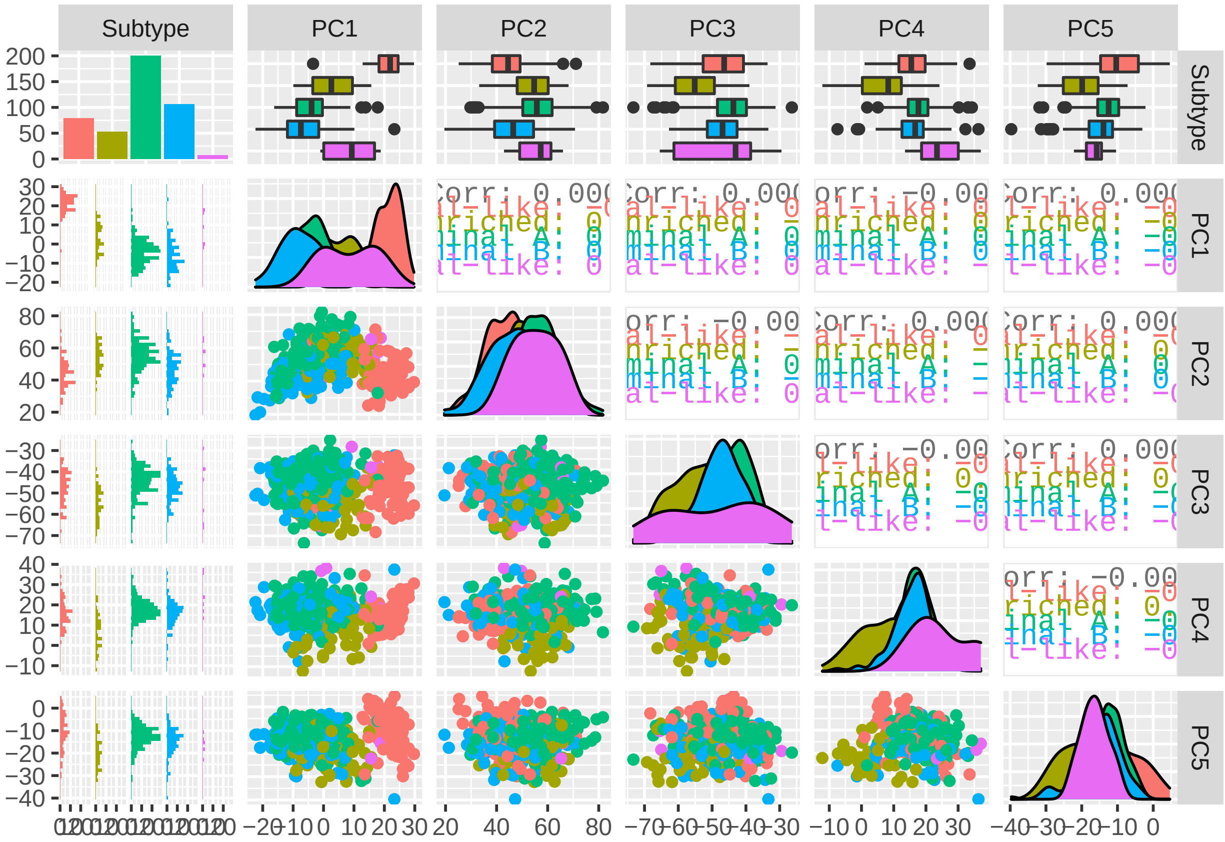
```
Cols=function(vec){cols=rainbow(length(unique(vec)))
                   return(cols[as.numeric(as.factor(vec))])}

sv = svd(scale(gdat,center=TRUE,scale=FALSE))
V = sv$v
Z = gdat%*%V

K = 3
pclabs = c("PC1","PC2","PC3","PC4")
par(mfrow=c(1,K))
for(i in 1:K){
  j = i+1
  plot(Z[,i],Z[,j],pch=16,xlab=pclabs[i],ylab=pclabs[j],col=Cols(cdat$Subtype))
}
legend(-45,0,pch=16,col=rainbow(5),levels(cdat$Subtype))
```



Pairs Plot

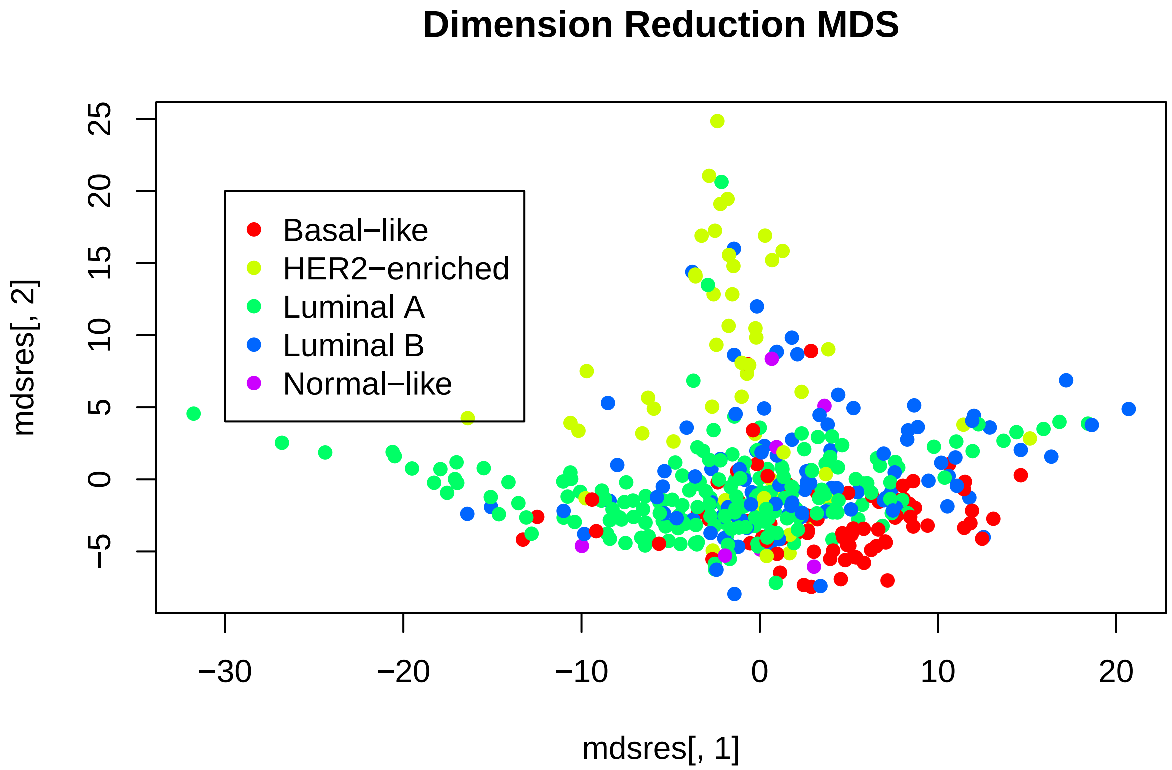
```
PC1<-as.matrix(Z[,1])
PC2<-as.matrix(Z[,2])
PC3<-as.matrix(Z[,3])
PC4<-as.matrix(Z[,4])
PC5<-as.matrix(Z[,5])

pc.df.cdat<-data.frame(Subtype = cdat$Subtype, PC1, PC2, PC3, PC4, PC5)
ggpairs(pc.df.cdat, mapping = aes(color = Subtype))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



MDS

```
Dmat = dist(gdat,method="maximum")
mdsres = cmdscale(Dmat,k=2)
plot(mdsres[,1],mdsres[,2],pch=16,col=Cols(cdat$Subtype), main = "Dimension Reduction MDS")
legend(-30,20,pch=16,col=rainbow(5),levels(cdat$Subtype))
```

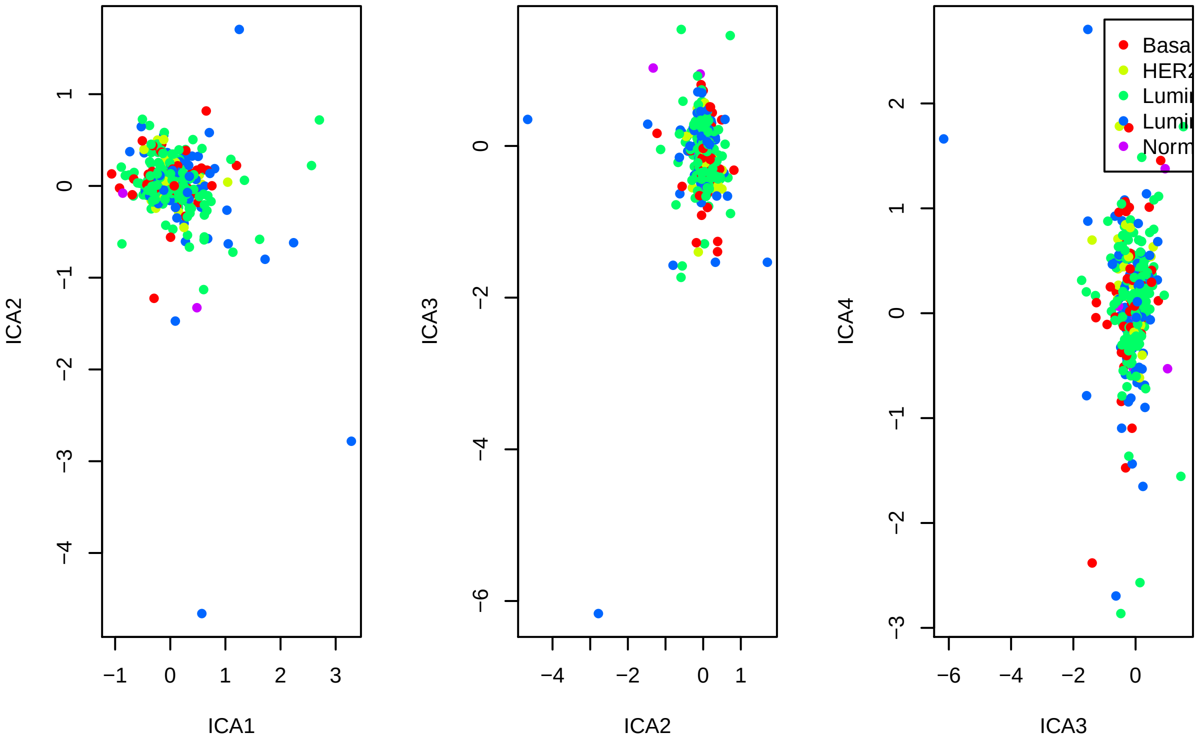


ICA

```r
require("fastICA")
```

```
## Loading required package: fastICA
```

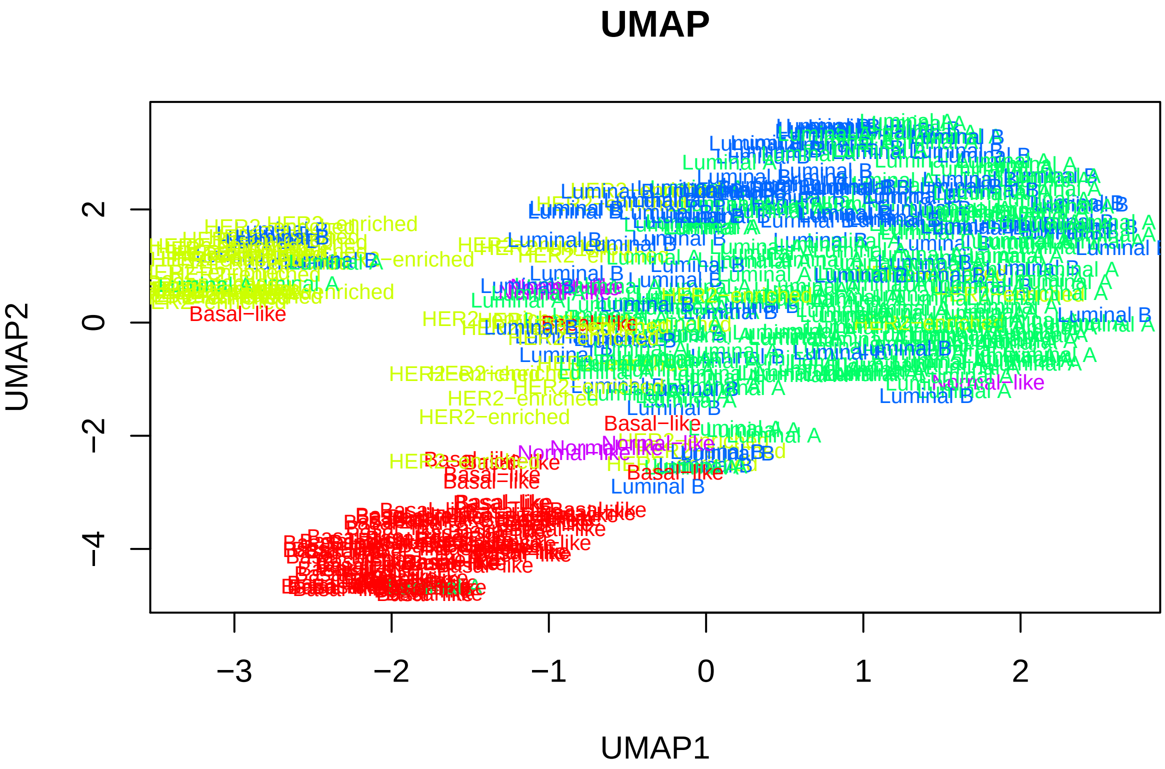
```r
K = 4
icafit = fastICA(gdat,n.comp=K)

kk = 3
pclabs = c("ICA1","ICA2","ICA3","ICA4")
par(mfrow=c(1,kk))
for(i in 1:kk){
  j = i+1
  plot(icafit$A[i,],icafit$A[j,],pch=16,xlab=pclabs[i],ylab=pclabs[j],col=Cols(cdat$Subtype))
}
legend(-1,2.8,pch=16,col=rainbow(5),levels(cdat$Subtype))
```



UMAP

```
gdat.umap = umap(gdat)
plot(gdat.umap$layout[,1], y =gdat.umap$layout[,2], type = "n", main = "UMAP", xlab = "UMAP1", ylab = "U
text(gdat.umap$layout[,1], y =gdat.umap$layout[,2], type = "n", cdat$Subtype, col=Cols(cdat$Subtype), c
```

```
## Warning in text.default(gdat.umap$layout[, 1], y = gdat.umap$layout[, 2], :
## graphical parameter "type" is obsolete
```



#Clustering

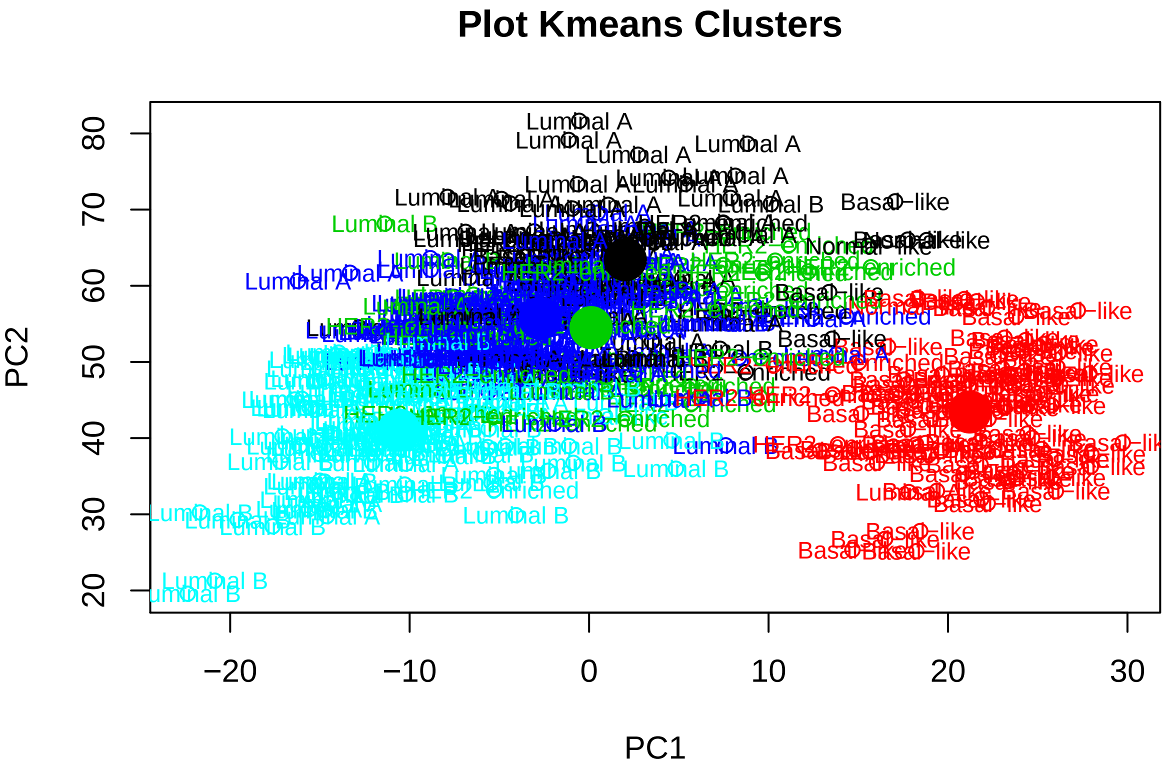
K-means

```
K = 5
km = kmeans(gdat,centers=K,nstart=25)
table(km$cluster,cdat$Subtype)
```

```
##
##     Basal-like HER2-enriched Luminal A Luminal B Normal-like
##   1          5             8        44        12           4
##   2         74             5         1         1           1
##   3          0            31         3         7           0
##   4          0             7       112        26           1
##   5          0             2        40        60           1
```

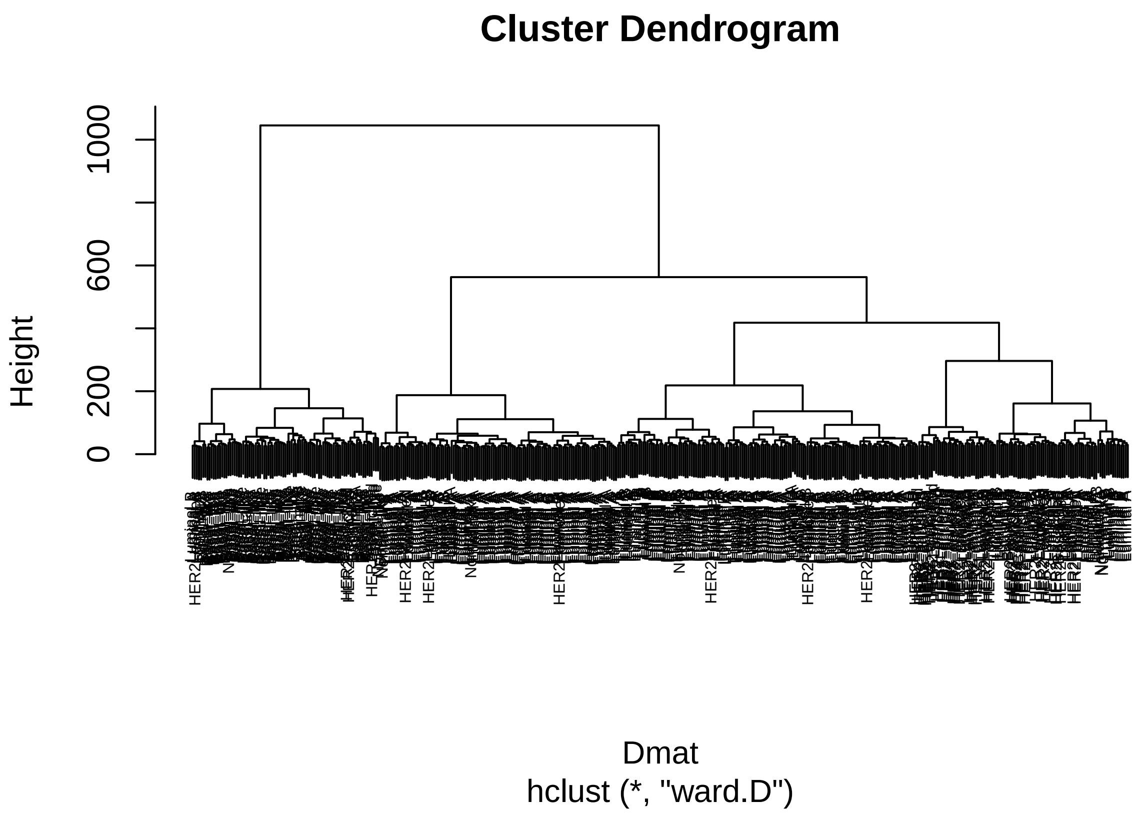
Plot Kmeans with labels

```
plot(Z[,1],Z[,2],col=km$cluster, main = "Plot Kmeans Clusters ", xlab = "PC1", ylab = "PC2")
text(Z[,1],Z[,2],cdat$Subtype,cex=.75,col=km$cluster)
cens = km$centers
points(cens%*%V[,1],cens%*%V[,2],col=1:K,pch=16,cex=3)
```



Hierarchical

```
#which linakge is the best?
#which distance metric is the best?

Dmat = dist(gdat)
com.hc = hclust(Dmat,method="ward.D")


plot(com.hc,labels=cdat$Subtype,cex=.5)
```



```
res.com = cutree(com.hc,5)
table(res.com,cdat$Subtype)
```

```
##
## res.com Basal-like HER2-enriched Luminal A Luminal B Normal-like
##       1          1             3        95        11           3
##       2          0             4        73        65           1
##       3         75             4         5         4           1
##       4          0            27         3         7           0
##       5          3            15        24        19           2
```

Consensus Clustering with Hierarchical

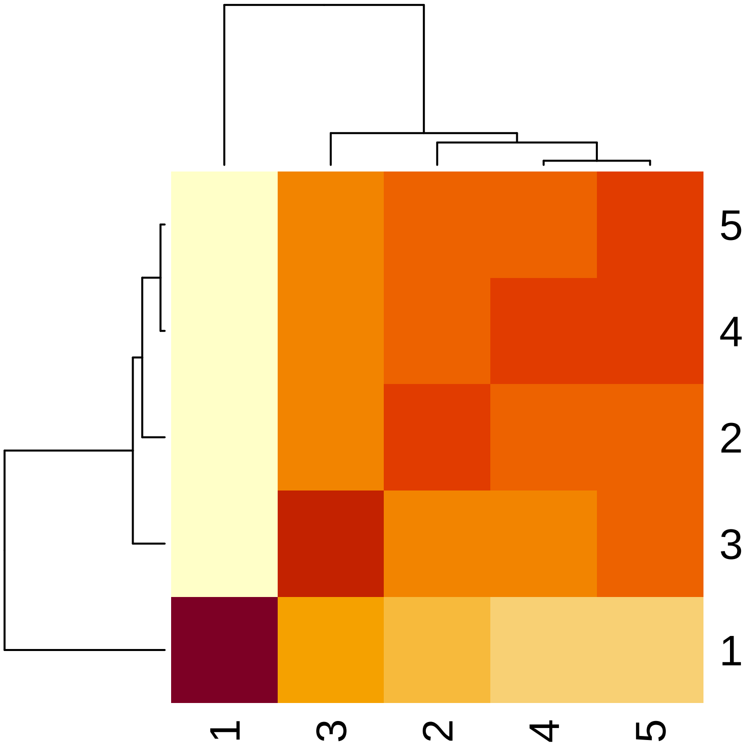
```
results = ConsensusClusterPlus(gdat,maxK=6,reps=500,pItem=0.8,pFeature=1,
clusterAlg="hc",distance="pearson",plot="png")
```

```
## end fraction
```

```
## clustered
## clustered
## clustered
## clustered
## clustered
```

Look at results for first 5 clusters

```
heatmap(results[[2]][["consensusMatrix"]][1:5,1:5])
```

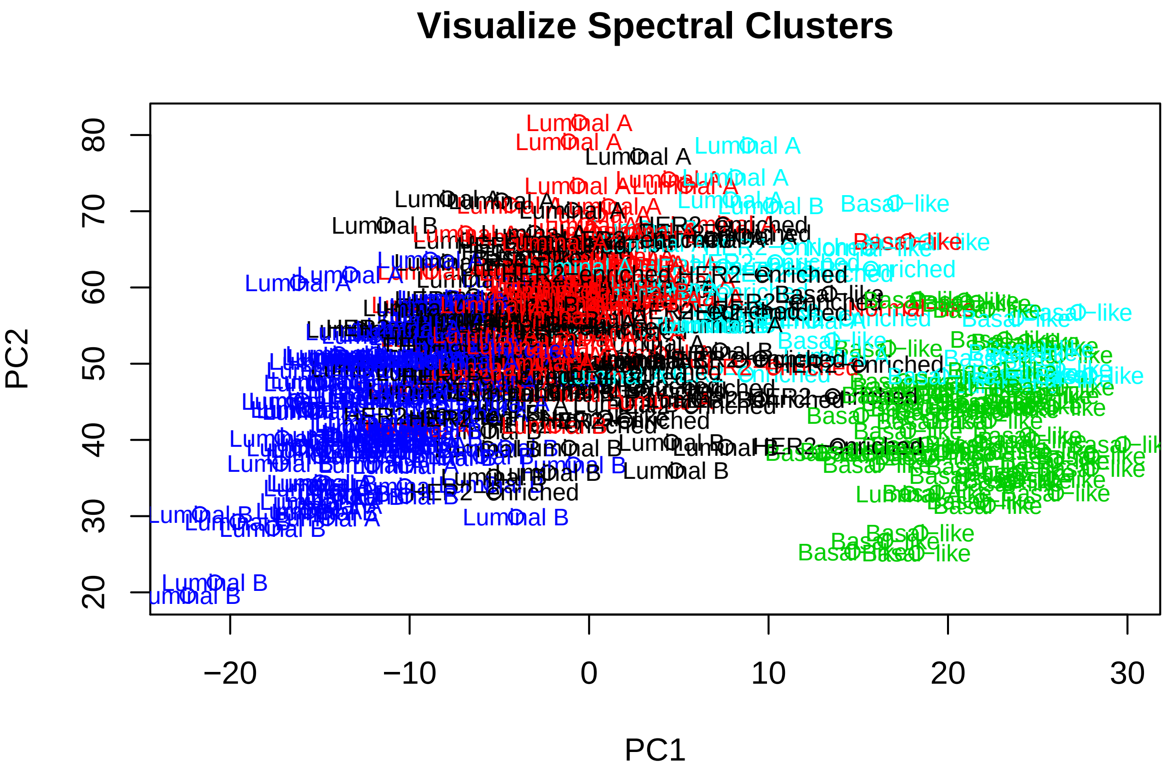


Spectral Clustering

```
K = 5
s_gdat = specClust(gdat, centers=K, nn = 7, method = "symmetric", gmax=NULL)
```

Visualize

```
plot(Z[,1],Z[,2],col=s_gdat$cluster, main = "Visualize Spectral Clusters", xlab = "PC1", ylab = "PC2")
text(Z[,1],Z[,2],cdat$Subtype,cex=.75,col=s_gdat$cluster)
```



Genes significantly associated with ER or PR Status, etc

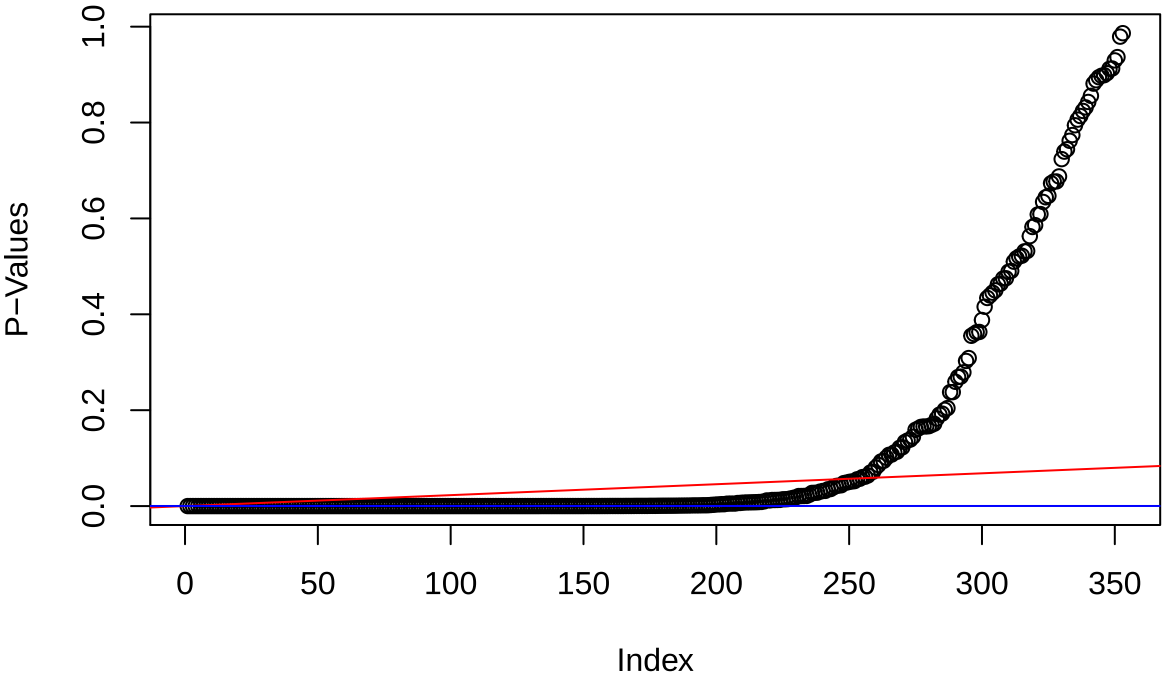
```r
x = gdat[cdat$ER=="Positive" | cdat$ER=="Negative",]
y.er = cdat$ER[cdat$ER=="Positive" | cdat$ER=="Negative"]
y.label = rep(1, length(y.er))
y.label[y.er == "Positive"]=2

ps = NULL
for(i in 1:ncol(gdat)) ps = c(ps,
 t.test(x[y.label==1,i],x[y.label==2,i])$p.value)
fdrs.bh = p.adjust(ps, method="BH")

cat("Number of Tests significant with alpha=0.1 using Bonferroni correction:",
sum(ps<0.1/length(y.label)), fill=TRUE)
```

```
## Number of Tests significant with alpha=0.1 using Bonferroni correction: 165
```

```r
cat("Number of Tests with FDR below 0.1:",
sum(fdrs.bh<0.1), fill=TRUE)
```

```
## Number of Tests with FDR below 0.1: 259
```

```r
plot(sort(ps,decreasing=FALSE),ylab="P-Values")
#BH procedure
abline(a=0, b=0.1/length(y.label),col="red")
#Bonferroni
abline(a=0.1/length(y.label), b=0,col="blue")
```



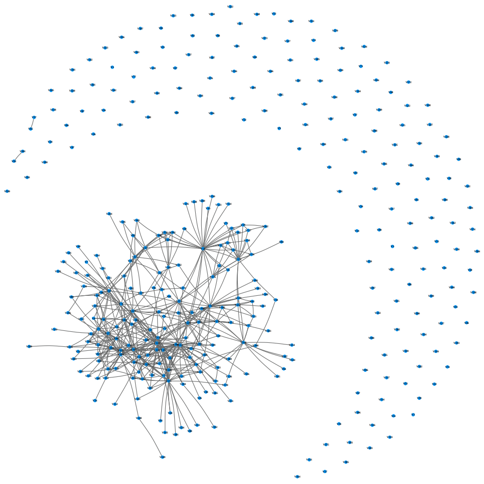
Graphical models - how are genes related?

```
#this takes a bit of time

lam = lambdaMax(gdat)*sqrt(log(ncol(gdat))/nrow(gdat))*0.01
net = XMRF(t(gdat),method="LPGM",lambda.path=lam,N=1,th=.0025)
```

```
## LPGM ::: Conducting sampling ... in progress:  100 %
## LPGM Completed.
```

```
plot(net)
```

```
## Plot optimal network
```



```
##                   [,1]        [,2]
##   [1,]  1.6355344883 27.85025999
##   [2,] 22.1442718477  4.14945599
##   [3,] 20.7325391667 15.62707111
##   [4,]  0.3598560423 10.72861394
##   [5,]  5.1391560551  6.80952548
##   [6,]  3.4422169535  7.05897462
##   [7,] -0.1282834254 14.56198277
##   [8,]  9.5211221830 14.02841568
##   [9,] 13.0120031442 23.67542195
##  [10,]  7.3526527684  7.57299682
##  [11,] -0.6999361967  8.46498822
##  [12,] 12.6524368851 31.49232595
##  [13,] 15.3526004893 29.68442138
##  [14,] 18.3112578970 16.69294316
##  [15,] 23.7452057943 15.96816252
##  [16,]  1.0395053223 24.91544632
##  [17,] 16.1281396299  0.27800071
##  [18,] 12.7739057304 11.98069403
##  [19,]  2.8573227816 14.64400438
##  [20,]  5.5804950852  4.27530664
##  [21,]  5.0686927237 29.21347417
##  [22,] 23.5885225347  6.23470186
##  [23,] 16.4248900544 21.03509625
##  [24,] -2.6084351886 22.75475378
##  [25,]  4.2226533269  9.94097810
##  [26,]  7.5784925085  5.28437187
##  [27,]  5.9206233502 27.79913018
##  [28,] 24.5952400448 20.27147894
##  [29,]  8.8551836752  9.53056592
##  [30,]  8.0147185082  5.70007630
##  [31,]  0.4531342332  5.11687151
##  [32,]  9.8772495984 15.38752244
##  [33,] 25.0270638272 11.50961393
##  [34,] 24.2595696269 14.16995406
##  [35,] 23.1327692716 20.23427271
##  [36,]  0.3443657267 23.29555877
##  [37,] 18.1738976934 24.60213176
##  [38,]  6.5466771640 16.83831716
##  [39,] 22.5617117435  5.27419366
##  [40,] 21.0985382263 19.89752113
##  [41,]  5.0942969586  8.56800782
##  [42,]  3.8874036855 15.53962816
##  [43,] 18.2052640547 20.63998893
##  [44,]  1.9397362829 10.14900066
##  [45,]  8.1633930409 18.22557646
##  [46,] 25.7895605083 10.42633628
##  [47,] -1.1159025759 22.37710522
##  [48,]  0.8827201102 12.50127593
##  [49,] -0.6924219515 15.61649468
##  [50,] 26.0177653066 14.01586359
##  [51,] 20.8060742491  4.07594192
##  [52,] 10.6121626727 24.26534039
##  [53,] 21.6173033651  6.27133936
##  [54,]  3.4594853156  8.10452981
##  [55,]  8.9302289180 14.31661088
##  [56,] 18.5726269366 18.34714539
##  [57,] -3.6985587235 24.42821428
##  [58,]  7.8126336438 15.47724606
##  [59,] 10.5453556205 17.09260802
##  [60,]  6.0121553724 24.75449840
##  [61,]  1.4094371593 12.60410787
##  [62,] 11.1737595869 11.87767645
##  [63,]  0.6451504885  8.28007292
##  [64,]  5.5007066431 12.23068504
##  [65,] -3.9320218053 23.63918190
##  [66,] -1.4918614707 23.88827957
##  [67,] 18.3682576202  5.63636809
##  [68,] 20.6183083388 26.13519340
##  [69,] 11.3425306906 28.54174756
##  [70,] 17.0874476532  3.53091091
##  [71,] 18.5791627064 27.91154127
##  [72,]  9.3769600816 17.20305914
##  [73,]  0.6887262754 10.00019828
##  [74,] 24.4816840752  7.41449948
##  [75,]  8.2979899652 11.56615772
##  [76,]  2.2411817578 11.71055191
##  [77,]  3.0935735090 27.18869478
##  [78,]  5.2052516208 10.02948225
##  [79,]  5.1472162610 10.85840509
##  [80,]  6.9181964665  3.04266924
##  [81,]  4.9557430721 30.51712792
##  [82,]  7.1104879112 29.99673042
##  [83,]  3.0428409212 16.36863824
##  [84,] 13.1639312085 15.93907290
##  [85,]  7.0828534808 31.39911672
##  [86,] 21.8865524120 13.04185493
##  [87,] -0.4194452525 24.86023051
##  [88,] -1.1552924225 12.18131486
##  [89,] 25.2473442252 21.57301658
##  [90,]  1.9077055506  9.33768160
##  [91,]  4.9122946732  3.73736625
##  [92,] -5.0715235450 21.44559765
##  [93,] 22.3666136531 18.81658058
##  [94,] 13.3261629948  7.27909549
##  [95,]  6.1400453715 11.09518916
##  [96,]  4.4600773327 12.80879221
##  [97,] 14.3833158677 22.28362374
##  [98,]  8.8212911931 30.00460976
##  [99,] 19.8328649857 16.75793563
## [100,]  7.7275969992 11.39202597
## [101,] 17.1367569264 19.37680545
## [102,]  4.6830622043 26.08403735
## [103,] 19.1837753835  1.88752236
## [104,]  2.2079863776  8.25217390
## [105,] 12.8763172165  6.77014233
## [106,] 11.3496739988 13.32198360
## [107,] 16.5328978270 24.32865643
## [108,] 11.2118417585 26.46872054
## [109,]  8.4705436427  8.89807478
## [110,] 14.7281207943 25.33764796
## [111,] 15.0624041923  1.07763955
## [112,]  1.8100519726  4.86738689
## [113,]  1.3795088679 13.48519428
## [114,] 19.9916391398  5.18164272
## [115,] 21.4128848908 23.90842725
## [116,]  6.6422499005 18.54721982
## [117,]  6.9245015835  8.06057554
## [118,] -2.3959057696 24.94802539
## [119,]  0.0001957319  6.80211319
## [120,]  1.8652565465  7.30766639
## [121,] 20.6728014180 14.06179531
## [122,] 21.7468496425 25.24031032
## [123,]  8.8733850655 18.48169428
## [124,]  4.6644937925  9.03443060
## [125,]  6.3517603727  6.77297285
## [126,] 14.7984909209 23.93639650
## [127,] 24.0634822692 10.38636691
## [128,] -0.3065526629 12.91727515
## [129,] 10.2295214204 12.10702202
## [130,]  4.3914415707 27.80038286
## [131,]  4.7434234171  9.40500027
## [132,]  0.1804790996  9.67221342
## [133,] 17.5183428174 22.90894327
## [134,] 18.8062242967 29.26717933
## [135,]  3.1782700265 14.91713559
## [136,]  9.5870443504  6.76809473
## [137,] 20.3337209918  6.67291846
## [138,] 26.2528521241 12.49614870
## [139,]  3.5742786991 12.45201359
## [140,]  0.5879649159 12.28946426
## [141,] -1.3463526741 15.19404134
## [142,]  0.0849464242 28.35872845
## [143,]  5.2226181507  2.94079702
## [144,] 13.9053917097  7.89670973
## [145,]  0.6684586974  8.91403871
## [146,]  2.3322930165 17.30007294
## [147,]  6.6380594733  9.39775576
## [148,]  3.4412098457  3.91238543
## [149,]  5.0579930037  1.25602923
## [150,] 25.1046242452 15.43759695
## [151,] -0.4713105572 10.67663269
## [152,]  1.1822849518  6.64624000
## [153,]  9.4900529425 15.87649007
## [154,]  9.5514307802 18.51933922
## [155,]  5.4665475651  6.45555431
## [156,] 11.1177146351 15.08249929
## [157,] 10.9040872049 16.72405300
## [158,] 10.8132438024 16.02761854
## [159,]  7.0102561130  8.65677230
## [160,] 24.7755574606 12.84750525
## [161,] 10.1900715006 16.59736306
## [162,] 24.8381102861 17.23258070
## [163,]  8.8158305746  7.36626692
## [164,] 14.2511229175 -0.07589935
## [165,] -1.6933224761 14.59219221
## [166,] 20.1621243885 21.25736258
## [167,] 13.3842652768  8.13522841
## [168,] 11.4890351326 31.45852254
## [169,]  1.5360510542  8.71388451
## [170,]  8.7524305283 10.51227430
## [171,]  8.6723848174  6.43842706
## [172,] 21.8319168432 26.83865747
## [173,]  4.8468313280  7.63641694
## [174,]  2.7428729107  8.88411485
## [175,]  9.7656940172 16.86715546
## [176,]  8.4853925402 13.55681771
## [177,] 10.9160128254 10.22626943
## [178,]  9.3261069947  6.17678977
## [179,] 18.8003609707 22.08760239
## [180,] 24.2991363768 18.81365435
## [181,] 23.7692157922 21.77275505
## [182,] 23.3993337195 17.57876598
## [183,] 11.5455004387 12.76944225
## [184,]  6.4484923864  6.22821976
## [185,] 15.5683938664 28.39796275
## [186,]  9.1211785947 11.46004598
## [187,]  8.4359606273 28.51880794
## [188,]  4.0737424124 24.44856075
## [189,] 23.3690004536 12.24304488
## [190,] -2.9770849275 21.37205150
## [191,] -0.0541263220 13.65972005
## [192,]  8.5966513566  3.30110663
## [193,] 17.8207276099  2.26317112
## [194,] 19.6271003464 26.98141968
## [195,]  7.5272255406 10.46957265
## [196,] 18.8752407374  7.18802818
## [197,] 14.3048047324 26.94029388
## [198,] 19.2958893906  9.38911174
## [199,] 16.0084700161 22.58808702
## [200,] 11.3992053492  8.86227689
## [201,]  3.3577314835  5.23569662
## [202,]  4.2108057076  5.96031561
## [203,]  3.2804575311 28.86152857
## [204,] 26.5091096984 15.29753095
## [205,] 13.0923793134 25.96569042
## [206,]  0.6316479655  6.63498487
## [207,]  1.3398738425  7.27235029
## [208,] 12.0583647687 12.36285088
## [209,] -4.0328558700  8.80697882
## [210,] -1.2700943633  8.77417058
## [211,]  7.4196336635  3.43746163
## [212,]  6.1875767642 11.89855285
## [213,]  6.2676928823  8.89999050
## [214,]  5.5576957890  8.02358946
## [215,]  4.8133914267 11.17009186
## [216,]  9.9401686401 27.57265536
## [217,] 20.6764280396 18.18445012
## [218,] 10.5696366961  9.08100904
## [219,]  4.8465211763 13.84015061
## [220,]  2.9559487445 10.32717285
## [221,]  5.2052976923 16.58072877
## [222,] 25.8485350201 19.74458057
## [223,]  3.0403407925  6.64149986
## [224,]  6.3420145588  3.27173600
## [225,]  3.0304663454 11.22297699
## [226,]  3.6813153479  6.58908573
## [227,]  9.8154199732 25.72694406
## [228,] 24.4098847401 23.10949544
## [229,]  6.7876496004 10.41514633
## [230,]  8.4408696400 19.03905089
## [231,] 20.3610684343 28.15902258
## [232,] 19.7327400095 11.09798447
## [233,]  0.9733061350 14.12296558
## [234,]  4.6600939218  8.57899382
## [235,]  5.4169684256 16.08698787
## [236,]  2.2680123826 29.89887751
## [237,]  3.0163216671 25.49843186
## [238,] 16.7756698453 25.81401499
## [239,] 17.1822079646 27.64173782
## [240,]  4.3419863347  6.84673093
## [241,] -1.0834874795 27.67141248
## [242,]  1.7125157641 26.28494204
## [243,] 11.8907570616 11.55707793
## [244,] 18.3662822160 26.21162081
## [245,] -2.5405012551 26.26914941
## [246,] 12.4101937306 27.56921985
## [247,]  3.0769233080  8.60313652
## [248,]  5.4757021452  7.20592354
## [249,] 19.8244693182 24.94419689
## [250,]  7.0631785691 11.12575122
## [251,]  9.6978922135  5.09944841
## [252,] 10.1905500248 11.63910089
## [253,] 11.2563506899 10.86208597
## [254,] 13.8757398607  8.90105099
## [255,] 20.1919732745  8.22463931
## [256,] 22.2961353625 15.52555887
## [257,] 19.5436452036 19.55586736
## [258,]  7.6316062185  7.06805801
## [259,] 19.1363703428 14.59846047
## [260,]  7.7643228415 18.74127332
## [261,]  0.2886605471 26.64701468
## [262,] 12.0711464223 16.99004069
## [263,]  2.1596388529 23.92024181
## [264,]  1.4189105441 17.85931313
## [265,]  8.6326964193  5.63279206
## [266,] 21.9923961445 21.17674853
## [267,]  1.0370343812 10.65481446
## [268,] 13.8009006611 30.98925169
## [269,] -0.0228504915  9.13239601
## [270,] 10.2245764904 14.74657215
## [271,]  6.4514609741 12.79267599
## [272,]  1.3455204070  9.70333507
## [273,] 12.0083361580 29.82534746
## [274,] 26.1893328029 16.80781307
## [275,] 16.2046791326  2.11428660
## [276,]  8.2972284805 27.11414225
## [277,] -0.8216622954 13.83121450
## [278,]  4.0519356359  8.21327424
## [279,] 16.8264878481 30.35270319
## [280,] 22.4036264310 11.05460524
## [281,]  5.9550930585  2.79617575
## [282,]  3.2308844147 10.61544345
## [283,] -1.0045174222  7.95977095
## [284,]  5.7387867986 16.58610933
## [285,]  5.4454984278 14.20609642
## [286,]  4.7891389569 16.38175381
## [287,]  2.4423546442 13.61053789
## [288,]  7.9132691568  8.23263935
## [289,]  2.4498186059 10.61960628
## [290,] 21.9385751859 17.20594953
## [291,] -4.1685616820 20.33505778
## [292,]  2.8856360514 12.77993562
## [293,] 25.1018488786  9.04643111
## [294,]  3.1165356509  7.72888863
## [295,] 10.1184535824 29.29158259
## [296,] -1.3892590838 11.10511276
## [297,] 19.3761862579 12.78561860
## [298,] 19.5026911997 23.50693004
## [299,]  7.5696534379  8.96591554
## [300,]  6.2053484004 26.42002004
## [301,] -4.4760725730 22.11601233
## [302,] 21.0438339395  9.95377095
## [303,] 12.2633228671  9.87695166
## [304,] 26.0290966623 18.27806637
## [305,] 17.5591635917  0.91436795
## [306,] 18.3581736551  4.30901864
## [307,]  3.5457273580 30.49750831
## [308,]  5.9709350608  8.93972773
## [309,]  9.5760758592  7.95584800
## [310,] 17.2768007198 29.14615148
## [311,] 22.8363045707 14.08470200
## [312,] -2.3183478071  9.97847575
## [313,]  5.7899659300 31.36180170
## [314,]  0.5746019148 15.00735005
## [315,]  0.4450411804 11.57228309
## [316,] 19.3603158457  3.43389467
## [317,] 13.5821519333 29.63544367
## [318,]  7.6248227359 25.89741172
## [319,] 23.6922704716  8.47414638
## [320,] -1.1031975861 26.23822876
## [321,] 20.5497615786  2.61642271
## [322,]  9.6762392608 31.98445190
## [323,]  1.1477923654 29.16997445
## [324,]  2.1989588067  8.54066138
## [325,] 21.0115445196 11.90200965
## [326,] 10.7766913791  6.93497889
## [327,]  9.0229464060 15.69911254
## [328,]  7.2058325060 18.67598871
## [329,] -2.0700458412 13.93506166
## [330,] -5.5368172139 19.39635369
## [331,]  5.0088550446 12.68500625
## [332,]  6.1519235535 14.35566233
## [333,]  3.9495042139  7.58076640
## [334,] 22.8531986670  9.62019607
## [335,] 20.8961886972 22.55928637
## [336,]  8.3963146741 24.70694144
## [337,]  6.8381455893 28.70677254
## [338,] 23.1362038928 25.26092391
## [339,] 21.6319464911  8.48591853
## [340,] 12.0133962589 24.88250930
## [341,] 22.5721598266 22.65201974
## [342,] 22.5692143481  7.44358515
## [343,] 10.3313594496 30.83249657
## [344,] 15.8056948532 26.91748469
## [345,] -0.5384514834  7.09681592
## [346,]  3.9323839849  9.24495932
## [347,] 23.2804904706 23.94454058
## [348,]  0.7117468085  7.81675119
## [349,]  3.2929499150 17.41189849
## [350,]  8.4181684536 31.45882447
## [351,] 15.2043937716 30.99013908
## [352,]  9.7166966281 10.44665006
## [353,] 13.7677379329 28.34287452
```

```
## estimation using spacejam
net2 = SJ(gdat)
net2 = net2$G
sjnet <- 1*(net2)[,,1]
sjnet = graph.adjacency(sjnet, mode="undirected")
plot(sjnet, vertex.size=0.1, vertex.label=NA)
```